A COMPUTER LOGGER FOR THE VZ-200/300

By Alex Johnson Jr.

Many hams have purchased the VZ-200 computer marketed by Dick Smith Electronics and the more recent VZ-300 model. Some have also taken advantage of the various projects and kits that allow the computer to be utilised for RTTY and CW.

Being the son of an amateur, I couldn't help but wonder, "why leave it there?". I also couldn't help but notice the time and trouble involved in keeping a log. Every time a contact is made and the callsign rang a bell, valuable time was lost flipping through log pages to track down who, where and when.

Problems were also observed during contests when, with each contact you make, you have to either mentally or physically flip through the log to see if the station has been worked before and, in the case of some contests, to see if the required time between duplicate contacts has elapsed.

So, if you have a computer handy in the shack, why not also use it to relieve the everyday drudgery of log keeping?

The program listed here is short (as log programs go) and written in BASIC so even the most cautious user can type it in without much trouble. It requires 24K of memory, an 80-column printer and a cassette recorder. The program is written specifically for the Dick Smith GP-100 dot matrix printer, but should work with most printers without any worries.

The program includes many "secrets, tricks and short-cuts" that I have discovered after working with the VZ for some time. These are used throughout the program to save memory, so please type the program in exactly as indicated in the listing (although you can leave out the spaces outside PRINT/INPUT statements) as the memory is balanced and juggled between string and memory needs.

To save you some counting, long stretches of spaces inside PRINT statements have been printed as (x spaces). When you encounter this, just type the number of spaces indicated by 'x'. When you encounter the term (rev), this indicates reversed text as used in program listing on the VZ-200/300. The printer used to list the program does not reproduce reversed text very well.

I have personally checked the final printout and provided the editor with corrections and modifications needed to ensure that the program works effectively (and your editor has taken great care to correct the detected errors for a bug-free printout — ED).

TRICKS & SECRETS

Some of the memory-saving tricks used in the program include:

1. Beep each time a key is pressed . . .

```
POKE 30862,90: POKE 30863,52
X=USR(0)
```

2. Small quick beep that can't be switched off (see 'beep off') . . .

```
POKE 28761,1: POKE 28671,32
```

The VZ technical manual discusses the Peizo on page 7. Any address from 26624 to 28671 decimal (6800 to 6FFF hex) will result in a beep when POKEd with these values.

```
Beep on: POKE 30779,0
Beep off: POKE 30779,1
```

3. Memory left . . .

```
POKE 30862,212: POKE 30863,39
```

String memory . . .

```
PRINT USR(X$)
```

RAM memory . . .

```
PRINT USR(X)
```

4. Sound abbreviations: the use of a semi-colon between notes and durations . . .

```
SOUND 16,2;21,7;15,3;22,8
```


1 of 5.

7 Oct. 1986. p 38-42.

5. THEN, GOSUB and GOTO on IF . . THEN . . ELSE statements: THEN can be replaced by a comma; GOTO can be left out; just the line number needs to be typed in after the comma. If a GOSUB is needed, then the comma can be left off . . .

IF A = B,100; IF A = B, PRINT "HI"
IF A = B GOSUB 100

6. REM can be replaced with an inverted comma and NEXT may be used with no variable . . .

100 ' THIS IS FUN
200 FOR A = 1 to X: NEXT

There are many more "tricks" that I did not use in the program. If you have others or want to know the rest, write to me at 19 Banksia St, O'Connor, ACT, 2601.

PROGRAM DESCRIPTION

The program is broken into subroutines and components that can easily be used to help iron out any bugs. The following is a brief summary of the subroutines and components.

Lines 10-90: In Line 10 the screen background color is set and addresses 30862 and 30863 decimal (788E and 788F hex) are POKEd with the addresses of the beep routine in ROM, 13392 decimal (3450 hex). Starting entry number is entered and variables are DIMensioned and set. T% in Line 40 controls how many entries can be kept in RAM at the one time. If you have more than 28K of RAM you can increase this. See Lines 1020 to 1080.

Lines 90-176: The screen is set up. Note the number of periods or dots in these lines, as they are crucial in the log PRINTing process.

Line 180: Commands are entered. This is what is referred to as the command line or command entry point.

Lines 190-310: The input at the command line is checked for a valid input. If the input is valid, the program goes to the appropriate subroutine; otherwise it returns to Line 170.

Lines 320-340: The cursor is moved to the correct position to fill in the entries. Before the command line, a click is produced by toggling the Peizo high then low. This is done by POKEing decimal 28671 (6FFF hex), the byte before the start of screen RAM. See previous text.

Lines 350-360: The entries are checked for length and cut down to the correct size. This is necessary as the PRINTing is dictated by entry length, as can be seen from Lines 430 and 860-870. This method is used to save memory.

Lines 368-420: Previous entries are scanned through from the latest to the first — i.e. backwards. If a previous contact has been made, the most recent contact is displayed.

Lines 430-435: If the continuous PRINTing mode is on, this subroutine is used to make the hard copy. The importance of length of entries can be seen here as entries are simply PRINTed one after the other without TABs.

Lines 440-500: This is the 'FIND' subroutine where callsigns are compared with the one specified in Line 140. If a match is found, you can continue the log or execute a further search.

Lines 510-540: This is the 'DELETE' subroutine where callsigns are compared with the one specified in Line 140. If a match is found, you can continue the log or execute a further search.

Lines 550-630: The 'SORT' routine allows the user to sort in two fields, by entry or callsign. If callsign is selected, all callsigns in string RAM are compared and sorted into alphabetical order in Lines 570-630. If entry number is selected, the data in string RAM is sorted into numerical order of entry number. Both formats are completed on three common nested loops. The decision as to which field is to be sorted is made in Lines 590 and 600.

Line 640: 'FILES FULL' subroutine. This is used when the maximum allowable string RAM is reached. It is marked by an indicator in the top left corner of the screen and a series of beeps. The beeps in Line 640 are produced from the one SOUND statement with semi-colons separating each note/duration pair. The files are considered full when the entry number is greater than T% — see Line 160. It is then necessary to SAVE the entries.

Lines 650-660: Continuous PRINTing mode is toggled on and off. When Z% = 1, it is on. When Z% = 0, it's off. See Lines 175 and 420-435.

Lines 670-720: Entries are displayed on the screen. While the entries are being displayed, pressing 'P' or the space bar will 'PAUSE' the screen, while 'S' will stop the list and return you to the main entry page.

Lines 730-870: Hard copy of the entries is made in this subroutine. The entries are PRINTed one after the other in Lines 860 and 870 and are not TABbed, as in Lines 430-435. Unlike continuous print, the pages are numbered and headed in columns. Entries are printed in the current (sorted) order — i.e. if a SORT by callsign has been carried out, the log will be printed in callsign order; if not, it will be printed in entry order. Stop and pause are similar to Lines 700-710.

Lines 880-920: Entries are SAVEd to tape in this subroutine in the current (sorted) order.

Lines 930-980: Previously sAVEd entries are LOADed from tape in this subroutine.

Lines 990-1000: The key beep is toggled on and off in this subroutine by PEEKing decimal 30779 (7836 hex), checking its value and adjusting Y%. When 30779 has a value of 0 and Y% = 1, the beep is off. If 30779 has a value of 1 and Y% = 0, the beep is on. If you are wondering why I used Y% to switch the indicator in Line 136 and didn't simply PEEK 30779, it isn't because I didn't think of it but rather because, for some reason or other, the value in 30779 is intermittently misread and therefore unreliable alone.

Lines 1020-1080: Memory left subroutine. This is useful if you have a computer with more than the basic 24K. By adjusting T% in Line 40, and keeping an eye on this subroutine, you can have more entries in string RAM at the one time. The amount of memory is calculated by calling a routine at decimal 10196 (27D4 hex). When USR(X) is used, the amount of free RAM in bytes is derived. USR(X$) derives the amount of string RAM in bytes. Line 170 is used to reset decimal 30862 and 30863 to the beep routine. See Line 10.

USING THE PROGRAM

Once the program is typed in and appears to be error free, SAVE it before you attempt to RUN it as it may contain an error that causes the program to crash and be lost. Once it is SAVEd, it is then safe to start as the SAVEd copy can be loaded and edited if the program crashes.

Once you type RUN and press RETURN, the first screen will ask you to enter the starting entry number. If this is the first time you are using the program, it will be '1'. If you are going to LOAD previously saved entries, just press RETURN with no number.

There will be a slight pause as the computer works itself out, then the main entry screen will appear with a beep. The cursor will be on the bottom of the screen — commands are entered from this point and nowhere else. If you wish to complete an entry, press RETURN. The cursor will then move up to the date — type it in the DDMMYY format as indicated (1st October, 1986 would be '011086'). Remember to put a zero in front of the number if it is a single digit (i.e. 4th is 04, March is 03).

An important point to note is not to go beyond the dotted markers for each parameter. If you do, the computer will automatically remove the extra characters but your screen will end up in rather a mess. If the screen does happen to get into a mess, just use the CLEAN command to reprint it.

When you have the date typed in, press RETURN and the cursor will move down to callsign. Type this in, not forgetting to press RETURN once you have finished. The cursor will then move down to the time. As with date, time should be entered in the HHMM format — i.e. 7 PM EST would be entered as 1900 (or 1100 UTC).

Be careful not to use semi-colons or commas in any of the entries, especially in 'remarks', as this will cause an error in the computer — 'extra ignored' or 'redo' — making life just a bit confused. If you do encounter such problems, forget the entry you were typing in and press RETURN until you hear the command line click. You can now see the reason for this click that cannot be turned off. Type 'CLEAN' as before and restart the entry.

If the screen is complete and correct, type 'ZZ' and press RETURN. This fills in the entry.

SPECIAL COMMANDS

Once you have a few entries in the log, you can have some real fun. Remember **commands can only be entered on the bottom line of the screen — the command line,** and it is only necessary to enter the first two letters of each command.

FIND is used to look through the log entries for a specified callsign. When the callsign is located, it will be displayed by filling in the details on the entry screen. You can then continue the search for further contacts by pressing 'F', or resume log entries by pressing 'C'.

DELETE removes an entry, placing a void on the callsign and removing all other information. You are asked for the entry number, so if you're unsure of the number, use DISPLAY or FIND to locate it.

SORT rearranges the entries in alphabetical order (press 'C' for callsign) or in entry order (press 'E' for entry). This is a BASIC program, so sorting does take a long time. Make yourself a coffee and have a break while it sorts.

RESTART simply starts the program over again. All entries are removed from the memories and all variables are reset, so SAVE your log before using RESTART.

DISPLAY prints all entries onto the screen. if you wish to pause while the entries are listing, press 'P' or the space bar. To stop the list completely, press 'S'. Once printing is complete, press 'C' to continue log entries.

CP or Continuous Print is used to keep a running hard copy of all entries as they are made. CP pages are not headed or numbered. When CP is activated, it is indicated on the command line. CP is deactivated by retyping CP on the command line.

PRINT makes a hard copy of the entire log on the printer. You must first enter the page length and the inter-page length. The page length must be more than six lines. Each page is numbered and headed and also has the date displayed on it. Stop and Pause commands are the same as for DISPLAY.

SAVE is used to send log details from RAM to tape. The SAVEd entries all have the same file number, so take note of the counter number on the datasette each time you SAVE.

LOAD is used to retrieve previously SAVEd files from tape.

MEMORY displays the amount of RAM, string memory and file space remaining.

BEEP turns the key beep on and off. When BEEP is activated it is indicated on the command line. To deactivate it, type BE again.

CLEAN is used to clear and reprint the screen if it becomes messy through an error.

ZZ enters the on-screen details in the log file.

I have two versions of the program: the one listed here (tape version) and another for VZs with a disk drive. The disk version is, I must admit, far superior in speed, capacity and commands. If difficulties are met typing this program in, and you would like a tape or disk copy of the working program, please drop me a line at the address mentioned earlier, with a stamped self-addressed envelope, and I will offer some suggestions.

PROGRAM LISTING

```
0010 POKE 30744,1: POKE 30862,80: POKE 30863,52: CLEAR
     10000:CLS
0015 PRINT "SERIAL: TPE2.310586": PRINT
0020 PRINT "COMPUTER LOG BOOK": PRINT "BY ALEX JOHNSON"
0030 PRINT "(C) COPYRIGHT 1986": PRINT@386,"STARTING ENTRY
     [space]NUMBER";
0040 T%=99: INPUT S%: IF S%)9999 OR S%(0, 10 ELSE CLS
0050 DIM I1$(10,T%), J$(10), K$(10): D$="000000": N%=-1
0060 Z$="[32 spaces]"
0065 Y$="....................."
0070 IF C%=1, C%=0: CLS: GOTO 80 ELSE N%=N%+1
0080 N$=MID$(STR$(N%+S%),2,LEN(STR$(N%+S%)))
0090 IF LEN(N$)(4, N$="0"+N$: GOTO 90
0100 X=USR(0): PRINT@0,Z$;Z$;Z$;
0110 PRINT@96,"ENTRY  ? ";N$: PRINT "DDMMYY  ? ";D$
0120 PRINT "CALLSIGN? VK....": PRINT "HHMM    ? ...."
0130 PRINT "RECD R/S ..": PRINT "SENT R/S .."
0140 PRINT "FREQ MHZ? .......": PRINT "MODE    ? ..."
0150 PRINT "QTH     ? ..........": PRINT NAME    ? .......
     ..."
0160 PRINT "REMARKS ? ................": IF N%)T%,GOSUB 640
0170 PRINT@448, "[8 spaces]?[22 spaces]";
```


3 of 5.

```
0175 IF Z%=1, PRINT@448, "[rev]CP"
0176 IF Y%=1, PRINT@451, "[rev]BE"
0180 PRINT@456, "";: INPUT A$: IF A$="", 320 ELSE A$=LEFT$
     (A$,2)
0190 IF A$="FI", 440
0200 IF A$="DE", 510
0210 IF A$="SO", 550
0220 IF A$="RE", RUN
0230 IF A$="DI", 670
0240 IF A$="PR", 730
0250 IF A$="CP", 650
0260 IF A$="SA", 880
0270 IF A$="LO", 930
0280 IF A$="ME", 1020
0290 IF A$="BE", 990
0300 IF A$="ZZ", 350
0305 IF A$="CL", C%=1: GOTO 70
0310 GOTO 170
0320 FOR A%=1 TO 10: PRINT@(A%*32)+104, "";: IF A%=1, INPUT
     D$: GOTO 340
0330 INPUT  I1$(A%,N%)
0340 NEXT: POKE 28671,1: POKE 28671,32: GOTO 170
0350 PRINT@0,"[rev]CHECKING": I1$(0,N%)=N$: I1$(1,N%)=D$
0351 I1$(2,N%)=I1$(2,N%)+Y$: I1$(3,N%)=I1$(3,N%)+Y$
0352 I1$(4,N%)=I1$(4,N%)+Y$: I1$(5,N%)=I1$(5,N%)+Y$
0353 I1$(6,N%)=I1$(6,N%)+Y$: I1$(7,N%)=I1$(7,N%)+Y$
0354 I1$(8,N%)=I1$(8,N%)+Y$: I1$(9,N%)=I1$(9,N%)+Y$
0355 I1$(10,N%)=I1$(10,N%)+Y$
0356 I1$(2,N%)=LEFT$(I1$(2,N%),6): I1$(3,N%)=LEFT$
     (I1$(3,N%),4)
0358 I1$(4,N%)=LEFT$(I1$(4,N%),2): I1$(5,N%)=LEFT$
     (I1$(5,N%),2)
0360 I1$(6,N%)=LEFT$(I1$(6,N%),7): I1$(7,N%)=LEFT$
     (I1$(7,N%),3)
0362 I1$(8,N%)=LEFT$(I1$(8,N%),10): I1$(9,N%)=LEFT$
     (I1$(9,N%),10)
0364 I1$(10,N%)=LEFT$(I1$(10,N%),16)
0366 IF N%=0, 420
0368 FOR A%=N%-1 TO 0 STEP -1: IF I1$(2,N%)=I1$(2,A%), 380
     ELSE NEXT
0370 GOTO 420
0380 PRINT@0,I1$(0,A%);"   [rev]FOUND": PRINT I1$(2,A%);
     "[space]"; I1$(1,A%);
0390 PRINT "[space]"; I1$(3,A%); "[space]"; I1$(9,A%)
0400 PRINT@13, "[rev]P)RINT D)ELETE"
0410 A$=INKEY$: IF A$="P", 420 ELSE IF A$="D", 100 ELSE 410
0420 IF Z%=1, 430 ELSE 70
0430 FOR A%=0 TO 10: LPRINT I1$(A%,N%);: IF A%<10, LPRINT
     "[space]";
0435 NEXT: LPRINT: GOTO 70
0440 PRINT@0, "CALLSIGN TO FIND ? VK....": PRINT@17,"";:
     INPUT B$
```


4 of 5.

```
0450 PRINT@0, "[rev]SEARCHING"; Z$: FOR A%=0 TO N%-1
0460 IF I1$(2,A%)=B$, 470 ELSE NEXT: GOTO 100
0470 PRINT@0, "[rev]FOUND";Z$: FOR B%=0 TO 10: PRINT@(B%
     *32)+106, I1$(B%,A%)
0480 NEXT: SOUND 0,2:  PRINT@6, "[rev]F)URTHER C)ONTINUE"
0490 C$=INKEY$: A$=INKEY$: IF A$="F", 500 ELSE IF A$="C",
     100 ELSE 490

0500 X=USR(0): PRINT@0, "[rev]SEARCHING";Z$: NEXT: GOTO 100
0510 PRINT@0, "";: INPUT "ENTRY TO DELETE ";A%: A%=A%-S%
0520 IF A%<0 OR A%>N%-1, 100
0530 PRINT@0, "[rev]DELETING";Z$: FOR B%=1 TO 10:
     I1$(B%,A%)="[space]": NEXT
0540 I1$(2,A%)="VOID": SOUND 0,3: GOTO 100
0550 PRINT@0, "SORT BY [rev]E)NTRY C)ALLSIGN"
0560 A$=INKEY$: IF A$="E" OR A$="C", X=USR(0): GOTO 570
     ELSE 550
0570 PRINT@0, "[rev]SORTING";Z$
0580 FOR A%=0 TO N%-1: FOR D%=0 TO 10: J$(D%)=I1$(D%,A%):
     NEXT B%
0590 FOR B%=A% TO N%-1

0600 IF A$="C", IF J$(2)<=I1$(2,B%), 630
0605 IF A$="E", IF J$(0)<=I1$(0,B%), 630
0610 FOR D%=0 TO 10: K$(D%)=I1$(D%,B%): I1$(D%,B%)=J$(D%)
0620 J$(D%)=K$(D%): NEXT D%
0630 NEXT B%: FOR D%=0 TO 10: I1$(D%,A%)=J$(D%): NEXT D%,A%
     : GOTO 100
0640 PRINT@0, "[rev]FILES FULL": SOUND 31,1; 31,1; 31,1;
     31,1; 0,5: RETURN
0650 IF Z%=0, Z%=1 ELSE Z%=0
0660 GOTO 100
0670 CLS: PRINT "[rev]P)AUSE S)TOP": PRINT: SOUND 0,1: FOR
     A%=0 TO N%-1
0680 FOR B%=0 TO 10: IF B%=6 OR B%=9, PRINT "[7 spaces]";:
     IF B%=9, PRINT "  ";
0690 PRINT I1$(B%,A%);" ": NEXT: PRINT: PRINT

0700 A$=INKEY$: IF A$="[space]" OR A$="P", X=USR(0): SOUND
     0,5: GOTO 700
0710 IF A$="S", 715 ELSE NEXT
0715 PRINT "[rev]C)ONTINUE"
0720 A$=INKEY$: IF A$="C", CLS: GOTO 100 ELSE 720
0730 PRINT@0, "";: INPUT "PAGE LENGTH"; L%: IF L%<7, 730
0732 PRINT@0, Z$: PRINT@0, "";: INPUT "INTER-PAGE LENGTH";
     I%
0734 PRINT@0, Z$: PRINT@0, "";: INPUT "PAGE NUMBER";P%:
     P%=P%-1
0736 PRINT@0, "SET UP PRINTER [rev]S)TART WHEN READY"
0740 A$=INKEY$: IF A$="S", X=USR(0): GOTO 750 ELSE 740
0750 PRINT@0, "[rev]PRINTING[rev off] [rev]P)AUSE S)TOP";
     Z$: GOSUB 760: GOTO 840
0760 P%=P%+1: LPRINT CHR$(14); "COMPUTER LOG BOOK";
     CHR$(15);
0770 LPRINT TAB(45); "PAGE "; P%
0780 IF P%>1, B%=4: GOTO 815
0790 D%=6: LPRINT " BY ALEX JOHNSON"; TAB(60);
     MID$(D$,1,2); "/";
0800 LPRINT MID$(D$,3,2); "/"; MID$(D$,5,2)
0810 LPRINT " (C) COPYRIGHT 1986)"
0815 LPRINT: LPRINT
0820 LPRINT "ENRY DATE   CALSGN TIME R  S  FREQ MD QTH
     [8 spaces]";
0830 LPRINT "NAME[7 spaces]REMARKS": LPRINT: RETURN
0840 FOR A%=0 TO N%-1: D%=D%+1
0843 B$=INKEY$: A$=INKEY$
0845 IF A$="[space]" OR A$="P", X=USR(0): SOUND 0,5: GOTO
     843
0848 IF A$="S", 100
0850 IF D%>L%, FOR D%=1 TO I%: LPRINT: NEXT: GOSUB 760
0860 FOR B%=0 TO 10: LPRINT I1$(B%,A%);: IF B%<10, LPRINT
     "[space]";
0870 NEXT: LPRINT: NEXT: GOTO 100
0880 CLS: PRINT "[rev]SAVE[rev off] FILENAME ";: INPUT C$"
0885 PRINT@5, "[rev]S)TART WHEN READY[rev off]";Z$
0890 A$=INKEY$: IF A$="S", X=USR(0): PRINT@4, "[rev]ING";Z$
     :GOTO 900 ELSE 890
0900 PRINT# "LOG.DATA.START",C$,S%,N%
0905 FOR A%=0 TO N%-1: A$="": FOR B%=0 TO 10: A$=A$+I1$
     (B%,A%): NEXT
0910 PRINT# "LOG",A$
0920 PRINT@98, A% C$: NEXT: GOTO 980
0930 CLS: PRINT "[rev]LOAD[rev off] FILENAME ";: INPUT C$"
0935 PRINT@5, "[rev]S)TART WHEN READY";Z$
0940 A$=INKEY$: IF A$="S", X=USR(0): PRINT@4, "[rev]ING";
     Z$: GOTO 950 ELSE 940
0950 INPUT# "LOG.DATA.START", A$ S%, N%: IF A$=C$, 960 ELSE
     950
0960 FOR A%=0 TO N%-1: INPUT# "LOG",A$
0961 I1$(0,A%)=MID$(A$,1,4): I1$(1,A%)=MID$(A$,5,10)
0962 I1$(2,A%)=MID$(A$,11,16): I1$(3,A%)=MID$(A$,17,20)
0963 I1$(4,A%)=MID$(A$,21,22): I1$(5,A%)=MID$(A$,23,24)
0964 I1$(6,A%)=MID$(A$,25,31): I1$(7,A%)=MID$(A$,32,34)
0965 I1$(8,A%)=MID$(A$,35,46): I1$(9,A%)=MID$(A$,47,56)
0966 I1$(10,A%)=MID$(A$,57,72)
0970 PRINT A% C$: NEXT
0980 SOUND 20,9: C%=1: GOTO 70
0990 IF PEEK (30779)=0, POKE 30779,1: Y%=0: GOTO 100
1000 POKE 30779,0: Y%=1: GOTO 100
1010 SOUND 0,5: PRINT@0, Z$: GOTO 100
1020 CLS: PRINT "[rev]MEMORY LEFT": POKE 30862,212: POKE
     30863,39
1030 PRINT: PRINT: PRINT T%-N%+1; "FILES LEFT"
1040 PRINT USR(X); "BYTES OF RAM FREE"
1050 PRINT USR(X$); "BYTES OF STRING RAM FREE": PRINT
1060 PRINT: PRINT: PRINT "[rev]C)ONTINUE"
1070 POKE 30862,80: POKE 30863,52
1080 A$=INKEY$: IF A$="C", C%=1: GOTO 70 ELSE 1080
```


5 of 5.